MTS: Logic Case Studies

Ian Jones
ianjones@umich.edu

Agenda

Why isn’t my message showing up?

Most behavioral science specialists are not programmers; often a tech person will be called in at some point to answer that question.

Even for a programmer, the answer isn’t always obvious.
Fixing Preview Errors

- Flagged errors are usually easy to fix. Just use the link in the Preview error list to take you back to the offending row in the Message Editor:

```
Click To Hide The Error List
ProcessingError:NameError During "select" on msgid (6) for : Gender in if#Gendre == "Female";
ProcessingError:SyntaxError During "select" on msgid (7) for : invalid syntax (, line 1 in if#Gender == "Male"

Section1
Hide/Show Text
```

Fixing Preview Errors

- Syntax errors are usually missing parens, missing close quotes, misplaced commas, used single equals sign, etc.
  - Gender = “Male”  # should be ==
  - (Age < 50 and Motivation >= 5  # missing close paren
Name errors are characteristics not in the dictionary (or misspelled or miscapitalized).

Notice the blue highlighting in the message editor:
- Gender == "Female"  # typo

<table>
<thead>
<tr>
<th>Text</th>
<th>Gender == &quot;Female&quot;</th>
<th>..</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>Gender == &quot;Female&quot;</td>
<td>..</td>
</tr>
</tbody>
</table>

However, at the moment the blue highlighting in the message editor will not catch incorrect capitalization:
- Stresslevel < 4  # should be StressLevel

<table>
<thead>
<tr>
<th>Text</th>
<th>StressLevel == &quot;Massive&quot;</th>
<th>..</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>StressLevel == &quot;Minimal&quot;</td>
<td>..</td>
</tr>
</tbody>
</table>
Fixing Preview Errors

Name errors can also be unquoted data values.

Since there’s no characteristic called “Female” in the dictionary (it’s looking for a characteristic, not a possible value), this will manifest as a Name Error

- Gender == Female # data value needs quoting
- Gender == “Female” # ok

My row’s not firing

- What should the characteristic value be at that point?
  - For debugging, make a Text row with a $Variable substitution as the message

- Is the value you’re testing against spelled correctly?
  - Gender==“Female” # shows error in Preview
  - Gender==“Femael” # no visible error

- Has a Select or List command reached its limit?

- Are you testing a string against a multiple-response list or vice versa?

- If a derived characteristic, is the calculation working?
  - Is the derived calculation superseded by an assigned test case value?
Refactoring Message Logic

- Simplifying complex message logic
- Or, how to help the message authors and keep your sanity

When does a message doc need help?

- Long, repeated expressions
- Expressions with complicated booleans or value comparisons
- Duplicated concepts
- Switches where a substitution would do
Refactoring long expressions

- If there's ambiguity, get the author to express in words what they mean. (Talk to them or have them write their intent into the Notes column.)

- Move clauses into Blocks if they apply to more than one row at a time

- Move clauses into Selects if they resemble a switch or case statement

Refactoring long expressions: example

```
Select 1:
  if BarrierTiming="Session4" and AdviceFirst="No" and CurrentIndBarScore>3 and CurrentIndBarScore<=5:
    you also feel you will
  if BarrierTiming="Session4" and AdviceFirst="No" and CurrentIndBarScore>4 and CurrentIndBarScore<=5:
    you also feel you might
  if BarrierTiming="Session4" and AdviceFirst="No":
    many people feel they might

if

Block if BarrierTiming="Session4":
  Select 1:
    if CurrentIndBarScore<6 and CurrentIndBarScore>=3:
      you also feel you will
    if CurrentIndBarScore<4 and CurrentIndBarScore=5:
      you also feel you might
    if CurrentIndBarScore<6:
      many people feel they might
```

University of Michigan Tailoring Workshop
Refactoring complicated booleans

- Move clauses into Blocks if they apply to more than one row
- Extract a clause into a derived characteristic
- Open a Python shell and try out the boolean

Refactoring complicated booleans: example

Extract a clause into a derived characteristic

Text if intersection(MaritalStatus, ['Married']) and QuitPartnerQuit=="Yes" and Gender=="Female": Don't fight with your husband...

Text if intersection(MaritalStatus, ['Married']) and QuitPartnerQuit=="Yes" and Gender=="Male": Don't fight with your wife...

# -----------------------------------------------

Derived:
IsMarried = intersection(MaritalStatus, ['Married'])

Text if IsMarried and nonSmoker and Gender=="Female": Don't fight with your husband...

Text if IsMarried and nonSmoker and Gender=="Male": Don't fight with your wife...
Demo: open a Python shell and try out the boolean

```python
((QuitSEControlHunger>=1 and QuitSEControlHunger<=3) or
 (QuitSEGainWeight>=1 and QuitSEGainWeight<=3) or
 QuitMainReasonStart="WeightGain") and QuitExerciseDocSaysNo="No"
```

Refactoring duplicated concepts

- If the concept appears in more than one place, especially across multiple message documents, extract the concept into a derived characteristic.

- Consider adding a reusable function to the project’s function library or to authorutil.py.
Refactoring switches

- Use the combination of a $Variable token in the message cell and a text substitution in the dictionary
- Also consider using a Select

Extra credit: catching errors early

- Write a script that checks the syntax of all the logic in all the messages
  - Easy using dictionary.py

- Expectations testing
  - For every test case I have, are the right rows firing?
Extra credit: expectations testing

Expectations testing

- Twin spreadsheets: test case definitions and expectations

<table>
<thead>
<tr>
<th>CASE</th>
<th>Gender</th>
<th>Health</th>
<th>PhyHealth</th>
<th>MentHealth</th>
<th>Reduced</th>
<th>Sleep</th>
<th>Energy</th>
<th>Diet</th>
<th>Diagnoses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>male</td>
<td>good</td>
<td>3</td>
<td>6</td>
<td>8</td>
<td>4</td>
<td>3</td>
<td>DH</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>female</td>
<td>good</td>
<td>3</td>
<td>2</td>
<td>8</td>
<td>5</td>
<td>3</td>
<td>dairy</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>male</td>
<td>good</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>4</td>
<td>3</td>
<td>none</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>female</td>
<td>good</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>4</td>
<td>3</td>
<td>none</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>male</td>
<td>good</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>4</td>
<td>3</td>
<td>none</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>female</td>
<td>good</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>4</td>
<td>3</td>
<td>none</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>Day2</th>
<th>Day3</th>
<th>Day4</th>
<th>Day5</th>
<th>Day6</th>
<th>Day7</th>
<th>Day8</th>
<th>Day9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>40</td>
<td>90</td>
<td>140</td>
<td>220</td>
<td>960</td>
<td>830</td>
<td>990</td>
<td>930</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
<td>100</td>
<td>150</td>
<td>230</td>
<td>980</td>
<td>840</td>
<td>980</td>
<td>930</td>
</tr>
<tr>
<td>3</td>
<td>60</td>
<td>110</td>
<td>170</td>
<td>240</td>
<td>980</td>
<td>860</td>
<td>980</td>
<td>930</td>
</tr>
<tr>
<td>4</td>
<td>50</td>
<td>100</td>
<td>150</td>
<td>230</td>
<td>980</td>
<td>840</td>
<td>980</td>
<td>930</td>
</tr>
<tr>
<td>5</td>
<td>60</td>
<td>90</td>
<td>150</td>
<td>250</td>
<td>770</td>
<td>840</td>
<td>980</td>
<td>930</td>
</tr>
<tr>
<td>6</td>
<td>50</td>
<td>100</td>
<td>170</td>
<td>270</td>
<td>980</td>
<td>840</td>
<td>980</td>
<td>930</td>
</tr>
</tbody>
</table>

Testing: higher-level gotchas

- Length
  - Too long or too short — easier to see later in development when you can compare a lot of test cases

- Readability